Product Feature Grouping for Opinion Mining

Zhongwu Zhai, *Tsinghua University*

Bing Liu, *University of Illinois at Chicago*

Jingyuan Wang, Hua Xu, and Peifa Jia, *Tsinghua University*

A constrained semisupervised learning method classifies words and phrases into feature groups, making it easier to produce an opinion summary of various product reviews.

Review aggregators and e-commerce sites are just two examples of businesses that rely on opinion mining to produce feature-based summaries of products' qualities.[1] This model first identifies product features and then collects positive and negative opinions on them to produce a summary of good and bad points.

Features of a product are attributes, components, and other aspects, such as "picture quality" and "zoom" for a digital camera.

Reviewers often use different words or phrases to describe the same product feature. For example, "picture" and "photo" mean the same thing for cameras. Grouping such synonyms is critical for effective opinion summary. Although WordNet and other thesauri can help to some extent, they aren't enough. For one thing, many words and phrases that are not synonyms in a dictionary may refer to the same feature in an application domain—for example, "appearance" and "design" are not synonymous, but they might both describe how something looks. For another, many synonyms are domain-dependent: "movie" and "picture" are synonyms in movie reviews but not in camera reviews, where "movie" is more likely to mean "video." And finally, determining which expressions indicate the same feature can depend on the user's application need. For example, in car reviews, internal design and external design can be two separate features or just one, called "design," according to the level of detail the user needs. In camera reviews, one shopper may want to study the battery as a whole, while another wants to know about battery weight, and battery life separately. For this reason, in applications the user needs to be involved in synonym grouping.

Formulating the Problem

Grouping feature expressions manually is time-consuming, because there are often hundreds of terms. This article describes a way to help the user perform the task more efficiently. Our approach assumes that feature expressions have been discovered from reviews by an existing system such as those described by Minqing Hu and Bing Liu and by Ana-Maria Popescu and Oren Etzioni.[1,2]

To reflect the user needs, he or she can manually label a small number of seed expressions for each feature group. The system then assigns the rest of the feature expressions to suitable groups. To the best of

1541-1672/12/$31.00 © 2012 IEEE
Published by the IEEE Computer Society

our knowledge, this approach has not been studied in opinion mining.[3,4]

We can formulate the problem as one of semisupervised learning.[5] The small set of seeds is the labeled data, and the rest of the discovered feature expressions are the unlabeled data. Any semisupervised-learning method can be applied. In this work, we use the expectation maximization (EM) algorithm. Specifically, we employ the naïve Bayesian EM formulation proposed by Kamal Nigam to solve our problem.[5]

However, we can do better, because the EM algorithm cannot ensure convergence to the problem's global optimum. What local optimum it achieves depends on the initialization—that is, the initial seeds L. Our approach shows that some prior knowledge can provide a better initialization and consequently generate better grouping results. Thus, we propose to create another set of data SL extracted from the unlabeled set U based on two pieces of natural language knowledge:

- Feature expressions that share some common words, such as "battery life" and "battery power," are likely to belong to the same group.
- Feature expressions that are synonyms in a dictionary, such as "image" and "picture," are likely to belong to the same group.

We call the newly created SL *soft-labeled examples* or *soft seeds*, and call the two pieces of prior knowledge *soft constraints* because they constrain the feature expressions to the same feature group. The constraints are soft (rather than hard) so that we can relax them in the proposed algorithm, which is important because they can create wrong groupings. In addition, when constraining the feature expressions in U with L to produce SL, conflict situations arise that need to be resolved. For example, if an unlabeled feature expression u in U is a single word, it may be synonymous with feature expressions in more than one feature group in L. Then, the question is which group u is likely to belong to. Given such conflicts, we also use lexical similarity to quantify the constraints' strength.

We conducted evaluations using reviews from five different domains. The results show that the proposed method outperforms the current state-of-the-art methods by a large margin.

Related Work in Synonym Grouping

Our work is primarily related to synonym grouping, which clusters words and phrases on the basis of their similarities. There are two main categories of similarity measures: those based on pre-existing knowledge resources (such as thesauri or WordNet) and those based on distributional properties.

In the first category, the work of Giuseppe Carenini and his colleagues is most related to ours.[6] They proposed a method for mapping feature expressions to a given domain feature taxonomy using several word similarity metrics. Our previous work also dealt with this problem relying on WordNet but with poor results.[7] Carenini's work and our previous work do not handle the domain dependency issue, for which a domain corpus is needed. In this article, we address the problem by using distributional contexts.

In the second category, we tested the k-means clustering with distributional similarity method discussed by Lillian Lee.[8] However, it does not perform well for our problem.

Recently, Fang Guo and his colleagues proposed a multilevel latent semantic association technique (mLSA) to group product feature expressions.[9] However, mLSA is unsupervised, and it cannot reflect users' needs. Another approach, multicorpus latent Dirichlet allocation (mLDA), runs LDA twice at two levels.[10] David M. Andrzejewski and his colleagues added two constraints to LDA by implementing the Dirichlet Forest prior, resulting in DF-LDA.[11] The two constraints are must-links and cannot-links. Must-links state that some data points must be in the same cluster, and cannot-links state that some data points cannot be in the same cluster. Experimental results show that our proposed method outperforms both these methods.

The Proposed Soft-Constrained Algorithm

The original input to the proposed algorithm consists of a set of reviews R and a set of discovered feature expressions F from R. According to the level of detail required, the user will first assign a small set of feature expressions L to the predefined feature groups C. The proposed soft-constrained EM algorithm SC-EM then assigns the rest of the discovered feature expressions U to C, as shown in Figure 1. The algorithm relies on the following equations:

$$P(w_t \mid c_j) = \frac{1 + \sum_{i=1}^{|D|} N_{ti} P(c_j \mid d_i)}{|V| + \sum_{m=1}^{|V|} \sum_{i=1}^{|D|} N_{mi} P(c_j \mid d_i)}, \quad (1)$$

$$P(c_j) = \frac{1 + \sum_{i=1}^{|D|} P(c_j \mid d_i)}{|C| + |D|}, \quad (2)$$

$$P(c_j \mid d_i) = \frac{P(c_j) \prod_{k=1}^{|d_i|} P(w_{d_i,k} \mid c_j)}{\sum_{r=1}^{|C|} P(c_r) \prod_{k=1}^{|d_i|} P(w_{d_i,k} \mid c_r)}. \quad (3)$$

Given a set of training examples D, each example d_i in D is considered an ordered list of words. In Equation 3, $w_{d_i,k}$ denotes the kth word in d_i, where each word is from the vocabulary $V = \{w_1, w_2, \ldots, w_{|V|}\}$. $C = \{C_1, C_2, \ldots, C_{|C|}\}$ is the set of predefined classes or groups, and N_{ti} is the number of times the word w_t occurs in d_i. To address the domain dependence issue mentioned before, we extract distributional contexts of each feature expression for SC-EM (we will provide the details later). That is, each example of L (or U) is actually the distributional contexts of the corresponding feature expression.

EM only achieves a local optimum, so we use natural language constraints to provide a better initialization—that is, to add more seeds that are likely to be correct, called SL (line 1 in Figure 1). For example, if one user-defined group L_i includes "appearance, exterior design," then SL_i could include "looking, design, exterior accessories." Because we extract SL using two soft constraints and lexical similarity, it is not guaranteed to be completely correct. So, we also allow SL to be violated in the learning process, as shown in lines 2–11. Compared with the original EM,[5] then, SC-EM has two main differences:

- In the first iteration (line 2), a set SL of soft-labeled examples (or soft seeds) is applied to initialize EM in addition to L. The training set size is increased, which helps produce better results.
- In the first iteration (line 2), the algorithm treats soft-labeled examples in SL the same way as the labeled examples in L. Thus, both SL and L are used as labeled examples to learn the initial classifier f_0. However, in the subsequent iterations (lines 4–11), SL is treated in the same way as any examples in U. That is, the classifier f_x from each iteration x (including f_0) will predict U. After that, a new classifier is built using both L and U_{PL} (U with probabilistic labels). Clearly, this implies that the class labels of the examples in SL are allowed to change.

```
    Input: Labeled examples L; Unlabeled examples U
    Output: Class labels of U
 1  Extract SL from U using constraints;
 2  Learn an initial naïve Bayesian classifier f
 3  using L ∪ SL and Equations 1 and 2;
 4  repeat
 5      // E-Step
 6      for each example d_i in U (including SL) :
 7          Compute P(c_j|d_i) using f and Equation 3.
 8      // M-Step
 9      Learn a new naïve Bayesian classifier f from L ∪ U
10      by computing P(w_t|c_j) and P(c_j) using Equations 1 and 2.
11  until the classifier parameters stabilize
12  Classify examples in U to C using the final classifier f.
```

Figure 1. The proposed soft-constrained expectation maximization algorithm. The algorithm assigns labels to a set of unlabeled feature expressions.

Finally, in line 12, the algorithm classifies the examples in U.

To continue the previous example, "exterior accessories" would be removed from C_i by SC-EM's revision, because its distributional contexts are different from the other feature expressions in $L_i \cup SL_i$. On the other hand, some unlabeled feature expressions would be newly grouped into C_i, such as "styling, style," by the trained classifier.

Extracting the Example Set Using Constraints

Figure 2 shows the detailed algorithm for extracting SL, the set of soft seed for providing a better initialization for SC-EM.

The algorithm includes an important subfunction, *isConstrained*(e_1, e_2) (lines 16–33), which checks whether two feature expressions e_1 and e_2 are affected by the proposed constraints. To match e_1 and e_2, we consider a few possibilities. If both e_1 and e_2 are single words (lines 18 and 19), the algorithm checks whether they are synonyms (line 20). If either e_1 or e_2 is a phrase, or if both of them are feature expression phrases, we see whether they have shared words.

In the main part of the algorithm (lines 1–15), L consists of a set of sets—that is, $L = \{L_1, L_2, \ldots, L_{|C|}\}$. Each L_i contains a set of labeled examples (feature expressions) of the ith class (feature group). Similarly, the output set SL also consists of a set of sets, $SL = \{SL_1, SL_2, \ldots, SL_{|C|}\}$. Each SL_i is a set of soft-labeled examples (feature expressions) of the ith class (feature group). Thus L_i and SL_i correspond to each other in that they represent the original labeled examples and the newly soft-labeled examples of the ith class, respectively.

In order to extract SL_i from U using L_i, we construct a graph (line 1) with feature expressions in U as nodes. The algorithm connects each pair of feature expressions (u_m, u_n) in U with an edge if u_m and u_n are constrained by *isConstrained* (lines 2–5). Experiments show that the graph G is very sparse. Then we can gather a set of connected components CC (line 6) without cutting any edges. Each component has two or more members, and all the members are likely to belong to the same feature group.

We then compare each connected component cc in CC with the labeled subsets $\{L_1, L_2, \ldots, L_{|C|}\}$ using lexical similarity based on WordNet (lines 7–15). The reason to use connected components rather than individual

```
1 Construct a graph G with all feature expressions in U as nodes
2 for each feature expressions u_m ∈ U do
3    for each feature expressions u_n ∈ U do
4       if m ≠ n & isConstrained(u_m, u_n) then
5          add edge(u_m, u_n) to G
6 Gather all connected components of G as CC
7 for each connected component cc ∈ CC do
8    for each feature group L_i ∈ L do
9       score(L_i) ← 0
10      for each feature expression e_1 ∈ cc do
11         for each feature expression e_2 ∈ L_i do
12            if isConstrained(e_1, e_2) then
13               score(L_i) ← score(L_i) + Sim(cc, L_i)
14               goto the loop in line 8
15   cc is added to SL_j such that arg max_{L_j} score (L_i)
16 // Function for checking constraints
17 isConstrained(e_1, e_2)
18   if e_1 is a single word expression then
19      if e_2 is a single word expression then
20         if e_1 and e_2 are synonyms then
21            return true
22      else // e_2 is a phrase
23         if e_1 ∈ e_2 then
24            return true
25   else // e_1 is a phrase
26      if e_2 is a single-word expression then
27         if e_2 ∈ e_1 then
28            return true
29      else // e_2 is a phrase
30         s ← e_1 ∩ e_2;
31         if |s| > 0 then
32            return true
33   return false
34 // Function for calculating similarity between sets
35 Sim(cc, L_i):
36   return Avg_{p1∈cc,p2∈L_i}(PrsSim(p1,p2))
37 // Function for calculating similarity between phrases
38 PrsSim(p1, p2):
39   return Max_{w_k ∈ p1, w_q ∈ p2} Jcn(w_k,w_q)
```

Figure 2. The algorithm for generating the soft-labeled set *SL*. The resulting "soft seeds" will provide a better initialization for SC-EM.

feature expressions as the basic units in comparison is that some words are not in WordNet. For example, for a pair u_m and u_n, if all members of L_i and u_m appear in WordNet while u_n does not, we cannot consider the similarity between L_i and u_n using WordNet. However, if u_m and u_n have been connected in a group using the constraints, the similarity between u_m and L_i can be approximated as the similarity between u_n and L_i. Thus we can consider u_n even though it is not in WordNet.

Intuitively, it is clear that if any members of *cc* are constrained with any members of L_i (lines 10–12 and 16–33), it means that *cc* is likely to belong to L_i and can be potentially added to SL_i. There are, however, conflict situations that need to be resolved—that is, *cc* may be constrained with more than one labeled subset L_i. For example, the single-word members of *cc* may be synonyms of feature expressions from more than one feature group. (Similar problems also occur when *cc*'s members are multiword expressions.) Then the question is which group *cc* should be assigned to.

Given these conflict cases, we use a score to record how accurate the grouping is. Once *cc* is compared with each labeled group L_i, the accumulated score is used to determine which class L_i has the strongest association with *cc*. We assign the class *j* with the highest score to *cc*; in other words, we add all members of *cc* to SL_j.

Regarding the score value, we used lexical similarity based on WordNet for this study. The pairwise score $Sim(cc, L_i)$ (line 13) is the average value of all the cross-similarities of their members (lines 34–36), computed by $PrsSim(p1, p2)$, where $p1 \in cc$ and $p2 \in L_i$. $PrsSim(p1, p2)$ calculates the WordNet similarity between two phrases (lines 37–39). In line 39, $Jcn(w_k, w_q)$ is the WordNet-based algorithm Jay Jiang and David Conrath proposed for calculating the similarity between two words w_k and w_q.[12] We also tried some other popular similarity measures, such as *Res* and *Lin*,[13,14] but *Jcn* performs the best for our task. These measures all rely on varying degrees of *least common subsumer* (LCS), which is the shared ancestor of the two concept words. For example, the LCS of *automobile* and *scooter* is *vehicle*. The similarity measure *Res* uses the information content of $LCS(w_1, w_2)$ as the similarity value as shown by the following two equations:

$$Res(w_1, w_2) = IC(LCS(w_1, w_2)) \text{ and}$$
$$IC(w) = -log\Pr(w),$$

where $\Pr(w)$ is the probability of the word w based on the observed frequency counts in the WordNet corpus. Both *Lin* and *Jcn* try to refine *Res* by augmenting it with the information content of w_1 and w_2 using the following two equations:

$$Lin(w_1, w_2) = \frac{2 \quad Res(w_1, w_2)}{IC(w_1) + IC(w_2)} \quad \text{and}$$

$$Jcn(w_1, w_2) = \frac{1}{IC(w_1) + IC(w_2) - 2 \times Res(w_1, w_2)}.$$

Distributional Context Extraction

To apply the proposed SC-EM algorithm, we need an example d_i for each feature expression e_i for naïve Bayesian learning. The example d_i (a so-called bag of words) is the aggregation of the distributional contexts of all sentences in L (or U) that contain the expression e_i. The context for e_i in a sentence s_{ij} is basically the surrounding words of e_i in a text window of $[-t, t]$, including the words in e_i. Stopwords are removed.

For example, e_i might be "screen," with two related sentences:

- s_{i1} = "The LCD screen gives clear picture."
- s_{i2} = "The picture on the screen is blurry."

We use the window size of [–3, 3]. In that case, s_{i1} gives us d_{i1} = <the, LCD, screen, gives, clear, picture>, and s_{i2} gives us d_{i2} = <picture, on, the, screen, is, blurry>. We then remove "on," "the," and "is" as stopwords, and we obtain the example d_i for e_i as a bag of words d_i = <LCD, screen, gives, clear, picture, picture, screen, blurry>.

Empirical Evaluation

To demonstrate the generality of the proposed method, we conducted experiments using reviews from five domains: home theater (HT), insurance (I), mattresses (M), cars (C), and vacuums (V). We obtained all the datasets and the feature expressions and their groups from a company that provides opinion mining services. Table 1 shows the details of the datasets.

Table 1. Datasets, feature expressions, and synonym groups.

	Domains				
Dataset size	**HT***	**I***	**M***	**C***	**V***
No. of sentences	6,355	12,446	12,107	9,731	8,785
No. of reviews	587	2,802	933	1,486	551
No. of feature expressions	237	148	333	317	266
No. of groups (thousands)	15	8	15	16	28

* HT = home theater, I = insurance, M = mattresses, C = cars, V = vacuums.

Evaluation Measures

Since SC-EM is based on semisupervised learning, we can use the classification *accuracy* to evaluate it. We can also evaluate how it clusters with initial seeds using the methods *entropy* and *purity*. (All three are standard measures for clustering.) In testing, the unlabeled set U is our test set.

Baseline Methods and Settings

We compared the proposed SC-EM with a set of existing methods, which can be categorized into unsupervised and semisupervised methods. The unsupervised methods include the following:

- *Collective hierarchical clustering* (CHC) applies complete-link hierarchical clustering to group feature expressions into k clusters based on the WordNet similarity as shown by lines 37–39 in Figure 2.
- *Simple hierarchical clustering* (SHC) is similar to CHC but uses single-link hierarchical clustering.
- *LDA* is a topic modeling method. Given a set of documents, LDA outputs groups of terms of different topics. In our case, each feature expression is a term, and each document refers to the distributional context of the feature expression.
- *mLSA*, based on LDA, is a state-of-the-art unsupervised method for solving the problem.
- *k-means* is based on distributional similarity with cosine as the similarity measure.

In the semisupervised category, we can further classify the methods into unconstrained, hard-constrained, and soft-constrained groups. The unconstrained subclass includes the following:

- *LDA*(L, H) is based on LDA, but the labeled examples L are used as seeds for each group (topic). All examples in L will always stay in the same topics. We call this hard initialization (H).
- *DF-LDA*(L, H) is the LDA method described before that takes must-links and cannot-links. We can express our L set as a combination of both kinds of links; unfortunately, we can apply only must-links because the number of cannot-links is huge and crashes the system. For example, for the car data, the number of cannot-links is 194,400 for 10 percent of the labeled data; for 20 percent, it is 466,560,000. DF-LDA also has a parameter η controlling the link strength, which we set very high (= 1,000) to reflect the hard initialization. We did not use DF-LDA in the unsupervised subclass, because without constraints it reduces to LDA.
- *K-means*(L, H) is based on k-means, but the clusters of the labeled seeds are fixed at initiation and remain unchanged.
- *EM*(L, H) is the original EM for semisupervised learning. Only the labeled examples are used as the initial seeds.

In the hard-constrained subclass, both our constraints are applied and cannot be violated. (LC is L plus SL produced by the constraints C.) It includes the following:

- *Rand*(LC, H) is an important baseline. It shows whether the constraints alone are sufficient to produce

good results. That is, the final result is the expanded seeds *SL* plus the rest of *U* assigned randomly to different groups.
- *LDA*(*LC*, H) is similar to LDA(*L*, H), but both the initial seeds *L* and the expanded seeds *SL* are considered as labeled examples. They also stay in the same groups throughout the process. Although *SL* is called a set of soft-labeled examples (seeds) in the proposed algorithm, we treat them as hard-labeled examples in this case just for experimental comparison.
- *DF-LDA*(*LC*, H) is DF-LDA with both *L* and *SL* expressed as must-links. Again, we use a large η (= 1,000) to make sure that must-links for *L* and *SL* will not be violated.
- *K-means*(*LC*, H) is similar to *k*-means(*L*,*H*), but both *L* and *SL* stay in their assigned clusters.
- *EM*(*LC*, H) is similar to EM(*L*, H), but *SL* is added to the labeled set *L*, and their classes are not allowed to change in the EM iterations.

In the soft-constrained (S) subclass, our two constraints can be violated. Initially, both the initial seeds *L* and the expanded seeds *SL* are considered as labeled data, but subsequently only *L* is regarded that way (that is, stays in the same classes). The algorithm re-estimates the label of each feature expression in *SL*. This subclass includes the following methods:

- *LDA*(*LC*, S), in contrast to LDA(*LC*, H), lets the *SL* change topics (groups).
- *K-means*(*LC*, S) is in contrast to k-means(*LC*, H).
- *SC-EM* can be grouped into this subclass.

We do not include soft-constrained DF-LDA in this subclass because different η values give different results, and they are generally worse than DF-LDA(*LC*, H).

For all LDA-based methods, the number of topics *K* is set to the number of groups (see Table 1). We used $\alpha = 50/K$ and $\beta = 0.01$, because Mark Steyvers and Thomas Griffiths found them to work well with many text collections.[15] The number of iterations was 1,000. We also experimented with different parameters and found these defaults performed well. For all *k*-means methods, the distance function was the cosine similarity.

Evaluation Results

We tested our SC-EM approach and the 16 baseline methods. To see the effect of different numbers of labeled examples (seeds), we experimented with using as the labeled set *L* 10, 20, 30, 40, and 50 percent of the feature expressions from the original data set, and the remainder as the unlabeled set *U*. All labeled data was selected randomly. For each setting, we ran the algorithms 30 times; Tables 2 and 3 show the average results. Because of space limitations, Table 2 shows the detailed accuracy, purity, and entropy results for only the runs with *L* comprising 30 percent of the labeled data (and 70 percent unlabeled). Table 3 summarizes the results for all other sizes of the labeled group *L*, averaged across the five domains. All the results were obtained from the unlabeled set *U*, which was also our test set. For the entropy measure, smaller values are better, but for purity and accuracy, the larger the better. For these experiments, we used the window size $t = 5$.

Tables 2 and 3 clearly show that the proposed algorithm outperformed all 16 baseline methods by a large margin on every dataset. In detail, we observe the following:

- SHC, CHC, LDA, mLSA and *k*-means with no seeds (no labeled data) performed the worst. Seeds helped improve the results, as expected. Without seeds, DF-LDA performs the same as LDA.
- LDA-based methods seems to be the weakest. K-means methods are slightly better, but EM methods are the best. This clearly indicates that classification (EM) performs better than clustering. DF-LDA's and *k*-means' results are similar.
- For LDA and *k*-means, hard-constrained methods (LDA(*LC*, H), and *k*-means(*LC*, H)) perform better than soft-constrained methods (LDA(*LC*, S) and *k*-means(*LC*, S)). This indicates that soft-constrained versions may move some correctly constrained expressions into the wrong groups. However, for the EM methods, the soft-constrained method (SC-EM) performs markedly better than the hard-constrained version (EM(*LC*, H)), indicating that the Bayesian classifier used in EM can take advantage of the soft constraints and correct some wrong assignments.
- Much weaker results of Rand(*LC*, H) than SC-EM in different settings clearly show that constraints alone (synonyms and sharing of words) are far from sufficient. EM can improve results considerably.
- Comparing the EM-based methods, we can see that soft seeds in SL make a big difference for all datasets. SC-EM is clearly the best.
- As the number of labeled examples increases (from 10 to 50 percent), the results improve for every method except DF-LDA, which does not change much.

We also varied the text window size *t* from 1 to 10 to see how it affected the performance of SC-EM. Figure 3 shows the results. It is clear that the window sizes of 2 to 5 produce similarly good results, so we used $t = 5$ for all our evaluations.

Table 2. Performance of feature expression classification and clustering methods on various domains (best performance in each category in bold).

		Accuracy					Purity					Entropy				
	Methods*	**HT**	**I**	**M**	**C**	**V**	**HT**	**I**	**M**	**C**	**V**	**HT**	**I**	**M**	**C**	**V**
Unsupervised	SHC	0.20	0.26	0.15	0.17	0.27	0.32	0.41	0.32	0.32	0.38	2.73	2.02	2.69	2.83	2.54
	CHC	**0.26**	**0.35**	**0.26**	**0.26**	**0.40**	0.37	0.46	0.33	0.32	0.43	2.47	2.04	2.56	2.63	2.09
	LDA	0.06	0.11	0.05	0.06	0.03	0.31	0.36	0.32	0.37	0.36	2.54	2.24	2.57	2.39	2.09
	mLSA	0.06	0.14	0.06	0.09	0.03	0.31	0.38	0.34	0.37	0.37	2.53	2.19	2.55	2.40	2.11
	***k*-means**	0.21	0.25	0.15	0.25	0.24	**0.42**	**0.45**	**0.39**	**0.44**	**0.47**	**2.14**	**1.90**	**2.32**	**2.16**	**1.78**
Semisupervised	LDA(*L*, H)	0.10	0.16	0.10	0.19	0.10	0.32	0.37	0.34	0.39	0.39	2.50	2.22	2.57	2.36	2.09
	DF-LDA(*L*, H)	0.27	0.25	0.19	0.28	0.31	0.37	0.41	0.39	0.45	0.40	2.32	2.00	2.35	2.15	1.98
	k-means(*L*, H)	0.20	0.25	0.17	0.27	0.20	0.42	0.43	0.42	0.48	0.48	2.12	1.92	2.26	2.04	1.76
	EM(*L*, H)	**0.48**	**0.50**	**0.52**	**0.56**	**0.49**	**0.50**	**0.53**	**0.56**	**0.58**	**0.52**	**1.93**	**1.69**	**1.87**	**1.80**	**1.79**
Hard-constrained	Rand(*LC*, H)	0.42	0.40	0.41	0.36	0.40	0.47	0.47	0.49	0.44	0.57	1.99	1.91	1.94	2.15	1.41
	LDA(*LC*, H)	0.47	0.43	0.45	0.47	0.43	0.53	0.50	0.52	0.56	0.59	1.81	1.79	1.84	1.76	1.41
	DF-LDA(*LC*, H)	0.35	0.33	0.23	0.34	0.37	0.49	0.49	0.39	0.51	0.52	1.86	1.71	2.26	1.88	1.58
	k-means(*LC*, H)	0.52	0.48	0.43	0.45	0.43	0.57	0.55	0.53	0.57	0.63	1.60	1.55	1.79	1.69	1.24
	EM(*LC*, H)	**0.61**	**0.62**	**0.54**	**0.63**	**0.53**	**0.64**	**0.65**	**0.60**	**0.66**	**0.63**	**1.41**	**1.33**	**1.57**	**1.38**	**1.26**
Soft-constrained	LDA(*LC*, S)	0.25	0.30	0.26	0.32	0.26	0.35	0.37	0.37	0.39	0.39	2.31	2.04	2.28	2.21	1.86
	k-means(*LC*, S)	0.33	0.30	0.29	0.29	0.33	0.45	0.43	0.46	0.45	0.53	1.99	1.94	2.10	2.09	1.58
	SC-EM	**0.68**	**0.72**	**0.68**	**0.75**	**0.68**	**0.68**	**0.74**	**0.70**	**0.76**	**0.69**	**1.30**	**1.07**	**1.26**	**1.12**	**1.16**

* See main article for meanings of acronyms and abbreviations.

Table 3. Performance of feature expression classification and clustering methods with varying percentages of labeled seeds.

		Accuracy					Purity					Entropy				
	Methods	**10%**	**20%**	**30%**	**40%**	**50%**	**10%**	**20%**	**30%**	**40%**	**50%**	**10%**	**20%**	**30%**	**40%**	**50%**
Unsupervised	SHC	0.20	0.21	0.21	0.22	0.26	0.33	0.35	0.35	0.37	0.38	2.72	2.63	2.56	2.46	2.25
	CHC	**0.25**	**0.28**	**0.31**	**0.34**	**0.37**	0.35	0.37	0.38	0.40	0.44	2.49	2.39	2.36	2.23	2.08
	LDA	0.07	0.07	0.06	0.06	0.08	0.33	0.33	0.34	0.35	0.38	2.50	2.44	2.37	2.28	2.11
	mLSA	0.07	0.07	0.08	0.07	0.07	0.34	0.35	0.35	0.37	0.38	2.48	2.42	2.36	2.26	2.12
	***k*-means**	0.22	0.23	0.22	0.22	0.22	**0.42**	**0.43**	**0.44**	**0.44**	**0.46**	**2.16**	**2.11**	**2.06**	**1.98**	**1.86**
Semisupervised	LDA(*L*, H)	0.10	0.10	0.13	0.14	0.15	0.34	0.34	0.36	0.37	0.39	2.48	2.43	2.35	2.25	2.11
	DF-LDA(*L*, H)	0.23	0.25	0.26	0.27	0.30	0.41	0.40	0.41	0.41	0.44	2.23	2.23	2.16	2.10	1.94
	k-means(*L*, H)	0.13	0.16	0.22	0.24	0.28	0.42	0.43	0.45	0.45	0.48	2.15	2.11	2.02	1.95	1.79
	EM(*L*,H)	**0.35**	**0.44**	**0.51**	**0.55**	**0.58**	**0.43**	**0.49**	**0.54**	**0.57**	**0.61**	**2.22**	**1.99**	**1.81**	**1.65**	**1.49**
Hard-constrained	Rand(*LC*, H)	0.33	0.37	0.40	0.44	0.46	0.40	0.47	0.49	0.54	0.57	2.32	2.00	1.88	1.64	1.50
	LDA(*LC*, H)	0.36	0.41	0.45	0.49	0.51	0.43	0.50	0.54	0.58	0.61	2.16	1.88	1.72	1.50	1.37
	DF-LDA(*LC*, H)	0.32	0.33	0.33	0.34	0.36	0.49	0.50	0.48	0.48	0.48	1.90	1.85	1.86	1.83	1.82
	k-means(*LC*, H)	0.38	0.43	0.46	0.51	0.53	0.44	0.54	0.57	0.61	0.62	1.93	1.69	1.57	1.40	1.32
	EM(*LC*,H)	**0.45**	**0.53**	**0.58**	**0.63**	**0.64**	**0.51**	**0.60**	**0.64**	**0.69**	**0.70**	**1.92**	**1.53**	**1.39**	**1.17**	**1.08**
Soft-constrained	LDA(*LC*, S)	0.18	0.23	0.28	0.32	0.36	0.36	0.39	0.37	0.46	0.50	2.37	2.25	2.14	1.90	1.74
	k-means(*LC*, S)	0.26	0.28	0.31	0.33	0.34	0.44	0.46	0.46	0.48	0.50	2.07	2.01	1.94	1.83	1.69
	SC-EM	**0.49**	**0.62**	**0.70**	**0.74**	**0.81**	**0.54**	**0.65**	**0.71**	**0.76**	**0.82**	**1.90**	**1.49**	**1.18**	**0.99**	**0.73**

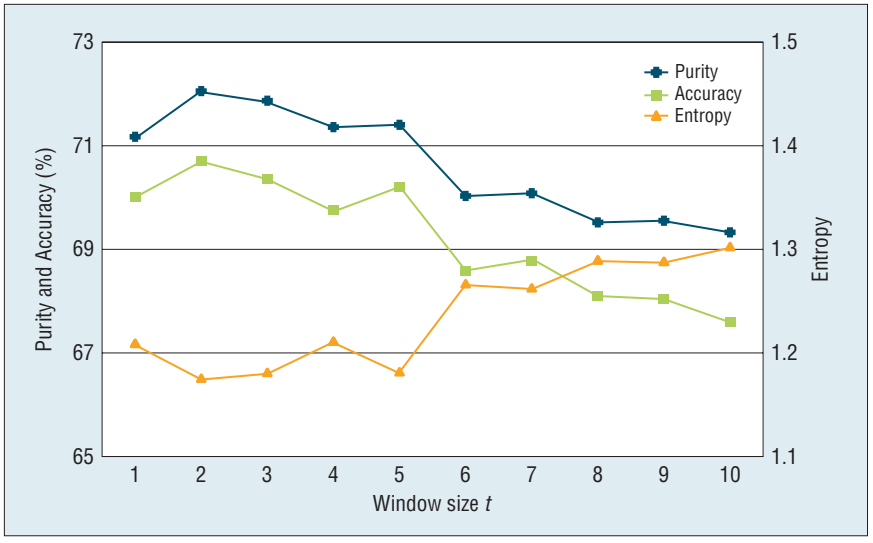

Figure 3. Influence of the context window size on SC-EM.

THE AUTHORS

Zhongwu Zhai is a PhD student at Tsinghua University. His primary research interests include sentiment analysis, opinion mining, and social media analysis. Contact him at zhaizw06@mails.thu.edu.cn.

Bing Liu is a professor of computer science at the University of Illinois at Chicago. His primary research interests include opinion mining and sentiment analysis, opinion spam (fake reviews) detection, Web mining, and data mining. Liu has a PhD in artificial intelligence from the University of Edinburgh. Contact him at liub@cs.uic.edu.

Jingyuan Wang is a PhD student at Tsinghua University. His research focuses on multimedia communication, over the Internet and wirelessly. Contact him at wangjingyuan06@mails.thu.edu.cn.

Hua Xu is an assistant professor at Tsinghua University. His current research interests include evolutionary computation, intelligent information processing, and advanced control technologies for circuit-manufacturing equipment. Contact him at xuhua@mail.thu.edu.cn.

Peifa Jia is a professor at Tsinghua University. His research interests include automatic control, information systems, intelligent robots, and assembly robot architecture. Jia has a PhD in computer science from Tsinghua University. Contact him at dcsjpf@mail.thu.edu.cn.

Empirical evaluations using five real-life data sets has shown that our proposed method is superior to 16 baseline methods. In our future work, we will focus on further improving the accuracy by exploiting more natural-language knowledge at the semantic level.

References

1. M. Hu and B. Liu, "Mining and Summarizing Customer Reviews," *Proc. ACM Int'l Conf. Knowledge Discovery and Data Mining* (SIGKDD 04), ACM, 2004, pp. 168–177.
2. A.-M. Popescu and O. Etzioni, "Extracting Product Features and Opinions from Reviews," *Proc. Conf. Empirical Methods in Natural Language Processing* (EMNLP 05), Assoc. for Computational Linguistics, 2005, pp. 339–346.
3. B. Pang and L. Lee, "Opinion Mining and Sentiment Analysis," *Foundations and Trends in Information Retrieval*, vol. 2, nos. 1–2, 2008, pp. 1–135.
4. B. Liu, "Sentiment Analysis: A Multifaceted Problem," *IEEE Intelligent Systems*, vol. 25, no. 3, 2010, pp. 76–80.
5. K. Nigam et al., "Text Classification from Labeled and Unlabeled Documents Using EM," *Machine Learning*, vol. 39, no. 2, 2000, pp. 103–134.
6. G. Carenini, R. Ng, and E. Zwart, "Extracting Knowledge from Evaluative Text," *Proc. 3rd Int'l Conf. Knowledge Capture* (K-CAP 05), ACM, 2005, pp. 11–18.
7. B. Liu, M. Hu, and J. Cheng, "Opinion Observer: Analyzing and Comparing Opinions on the Web," *Proc. 14th Int'l Conf. World Wide Web* (WWW 05), ACM, 2005, pp. 342–351.
8. L. Lee, "Measures of Distributional Similarity," *Proc. 37th Ann. Meeting Assoc. Computational Linguistics*, Assoc. for Computational Linguistics, 1999, pp. 25–32.
9. H. Guo et al., "Product Feature Categorization with Multilevel Latent Semantic Association," *Proc. 18th ACM Conf. Information and Knowledge Management* (CIKM 09), ACM, 2009, pp. 1087–1096.
10. D. Blei, A.Y. Ng, and M.I. Jordan, "Latent Dirichlet Allocation," *J. Machine Learning Research*, vol. 3, 2003, pp. 993–1022.
11. D. Andrzejewski, X. Zhu, and M. Craven, "Incorporating Domain Knowledge into Topic Modeling via Dirichlet Forest Priors," *Proc. 26th Int'l Conf. Machine Learning* (ICML 09), Omnipress, 2009, pp. 25–32.
12. J. Jiang and D. Conrath, "Semantic Similarity Based on Corpus Statistics and Lexical Taxonomy," *Proc. Int'l Conf. Research on Computational Linguistics* (ROCLING X); http://arxiv.org/pdf/cmp-lg/9709008.pdf.
13. P. Resnik, "Using Information Content to Evaluate Semantic Similarity in a Taxonomy," *Proc. 14th Int'l Joint Conf. Artificial Intelligence* (IJCAI 95), AAAI, 1995, pp. 448–453.
14. D. Lin, "An Information-Theoretic Definition of Similarity," *Proc. 15th Int'l Conf. Machine Learning* (ICML 98), Omnipress, 1998, pp. 296–304.
15. M. Steyvers and T. Griffiths, "Probabilistic Topic Models," *Handbook of Latent Semantic Analysis*, T. Landauer et al., eds, Laurence Erlbaum, 2007, pp. 424–440.